Development of a Process Assessment Model for Medical Device Software Development

Marion Lepmets, Paul Clarke, Fergal McCaffery, Anita Finnegan, Alec Dorling
Regulated Software Research Centre, Dundalk Institute of Technology,
Dundalk, Ireland
{marion.lepmets, paul.clarke, fergal.mccaffery, anita.finnegan, alec.dorling } @dkit.ie

Abstract

Software that is incorporated into a medical device, or which is a standalone medical device in its own right, is of a safety critical nature and subject to regulation from various jurisdictions (principally the EU and the US). In order to satisfy jurisdictional regulations, developers of medical device software generally implement standards and guidance provided by international standards bodies and national administrative departments. However, the various standards and guidance documents are not developed as a single cohesive set but often as separate resources addressing distinct areas of concern. The result for medical device software developers is that integration of these various sources represents a challenging undertaking.

The aim of this paper is to describe the integration of the several process models and regulatory standards, first, into a process reference model and then into a process assessment model for medical device software development. The focus is on the integration of regulatory standards from the medical device domain with generic software development process models, resulting in a unified best practice framework for medical device software development. The process reference model for medical device software development is going to be published this year as IEC TR 80002-3, and the process assessment model for medical device software development is currently being validated through pilot studies in medical device industry.

This best practice framework will help small software developers in their adoption of regulations-compliant best practices while reducing the overhead associated with understanding the long list of regulations and standards they need to adhere to when developing software for medical devices. This framework will also help the manufacturers in selecting their software suppliers assuring that the suppliers have adopted the best practices and are compliant with the medical device standards and regulations.

Keywords

Process Assessment Model, Process Reference Model, Medical Device Software, Medical Device regulations, Best Practice Framework

1 Introduction

A basic requirement of the design of a medical device software process is that it satisfies the regulatory demands associated with the medical devices under construction; as failure to satisfy regulation in a particular region will mean that the device cannot be placed upon the market. In a sense, this is similar to the basic requirement of a software development process: that the process should fit the needs of the project [1]. However, in practice the realisation of this basic requirement has proven to be difficult to achieve in the general software engineering domain. Consequently, many software development approaches have been proposed and implemented, resulting in much discussion regarding the benefits and limitations of the various approaches [2]. It therefore seems likely that no single software approach should be universally prescribed for the general practice of software development. This is essentially owing to the complex interplay between people and the economic activity of commercial software development.

1.1 Medical device regulations and standards

A medical device can consist entirely of software or have software as a component of the overall medical device system. In order to be able to market a medical device within a particular region it is necessary to comply with the regulatory demands of that region. Two of the largest global bodies responsible for issuing and managing medical device regulation belong to the central governing functions of the US and EU. In the case of the US, the Food and Drug Administration (FDA) issues the pertinent regulation through a series official channels, including the Code of Federal Regulation (CFR) Title 21, Chapter I, Subchapter H, Part 820 [3]. Under US regulation, there are three medical device safety classifications: Class I, Class II and Class III. The medical device safety classification is based on the clinical safety of the device. Class I devices are not intended to support or sustain human life, and may not present an unreasonable risk of harm. Class II devices could cause damage or harm to humans. An example of a Class II medical device is a powered wheelchair. Class III medical devices are usually those that support or sustain human life, and are of significant importance in the prevention of human health impairment. An example of a Class III device is an implantable pacemaker. All implantable devices are Class III medical devices as the surgery required carries with itself additional high risks from anaesthesia and possible infections that go beyond the technical and engineering safety risks of the correct performance of the device.
In the EU, the corresponding regulation is outlined in the general Medical Device Directive (MDD) 93/42/EEC [4], the Active Implantable Medical Device Directive (AIMDD) 90/385/EEC [5], and the *In-vitro* Diagnostic (IVD) Medical Device Directive 98/79/EC [6] - all three of which have been amended by 2007/47/EC [7]. Although slightly different to the US safety classifications that are based on clinical safety of the device, the EU classifications essentially embody similar classifications and limitations, where Class I corresponds to Class I, Class IIa and IIb to Class II, and Class III to Class III. A further safety classification applies to the software in the medical device as outlined in IEC 62304, wherein the safety classification is concerned with the worst possible consequence in the case of a software failure (as compared with general medical device safety classification which is based on the difficulty of a regulator to determine if the device will be safe). Hence, some Class II medical devices can cause serious injury or even death, but they are Class II because they are similar (in clinical use and safety) to well understood devices that have been used before. Since IEC 62304 safety classifications are based on worse case failure of the software, it is possible that Class II medical devices can have Class III software.
In the medical device domain, ISO 13485:2003 (ISO 13485 from hereon) [8] outlines the requirements for regulatory purposes from a QMS perspective. ISO 13485, which is based on ISO 9001 [9], can be used to assess an organisation's ability to meet both customer and regulatory requirements. However, ISO 13485 does not offer specific guidance on medical device software development. IEC 62304:2006 (IEC 62304 from hereon) [10], which can be used in conjunction with ISO 13485, does offer a framework for the lifecycle processes necessary for the safe design and maintenance of medical device software. As a basic foundation, IEC 62304 assumes that medical device software is developed and maintained within a QMS such as ISO 13485, but does not require an organisation to

be certified in ISO 13485. Therefore, IEC 62304 can be considered to be a software development specific supplement to ISO 13485.
IEC 62304 is based on ISO/IEC 12207:1995 [11] which although a comprehensive standard for software development lifecycle processes has effectively been decommissioned following the publication of the more extensive ISO/IEC 12207:2008 [12]. Furthermore, other developments in the ISO and IEC communities for software development, such as ISO/IEC 15504 [13], have provided significant additional levels of software process detail to support ISO/IEC 12207:2008. IEC 62304 is currently being revised to better align with ISO/IEC 12207:2008. IEC 62304 is a critical standard for medical device software developers as it is the only standard that provides recommendations for medical device software implementations based on the worst consequences in the case the software failure causing hazards. Furthermore, for general medical device risk management, IEC 62304 is used in conjunction with ISO 14971 [14], with IEC 80002-1 [15] providing guidance on the application of ISO 14971 for software development. Additionally, as IEC 62304 considers a medical device system to consist of software as part of an overall medical device system, the system level requirements are not included within IEC 62304 but instead within the medical device product standard IEC 60601-1 [16]. Also it should be noted that due to the increasing importance of usability within the medical device industry organisations should also adhere to the medical device usability requirements outlined in IEC 62366 [17].
Numerous different medical device standards and regulations now exist, some of which are interlinked with generic software development standards and others which are inconsistent. The dominant medical device software standards such as IEC 62304 are not yet aligned with the approach adopted in the general software development standards community since the 1995 publication of ISO/IEC 12207. One significant change in this respect has been the introduction of a harmonised approach to process description (as defined in ISO/IEC 24774 [18]) which involves the identification of core process outcomes that can later be harnessed to develop a process assessment method. A further significant change relates to the movement in the general software development standards community (and in other safety-related domains) to include a software process improvement dimension that can be instrumental in guiding software development organisations towards the required process targets. In effect, the medical device standards have not kept up with the changes that have been made to the general software development standards. There are several reasons why the medical device standards lag the updates to the general software development standards, (perhaps) most importantly the IEC stability period during which adopted harmonised standards are not to be changed unless the proposed changes are necessary in terms of safety. With the expanding role of software in medical devices, there is a case to be made for introducing the accumulated up-to-date wisdom in the general software development standards into the medical device software development specific standards in a uniform fashion – and work in this direction should not wait for the IEC stability period to come to an end, but rather proceed in the interim period (such as the work reported upon in this paper).
In order to identify an appropriate architecture for introducing the significant body of general software process knowledge into the medical device process domain, an initial important step involves the building of a process reference model (PRM) for medical device software development. The approach used for the PRM development is described in the next section. We then illustrate the architecture and the challenging task of integrating various regulatory standards and informative guidance into a process assessment model (PAM) to allow consistent evaluation of medical device software development processes against the set of standards these organisations have to adhere to. Finally, we summarize the paper along with some concluding remarks.

2 Development of the PRM

A process reference model (PRM) describes a set of processes in a structured manner through a process name, process purpose and process outcomes. Process outcomes are the normative requirements the process should satisfy to achieve the purpose of the process. The PRM for medical device software development is based on various international medical device and generic software development standards as shown on Figure 1.
The development of the PRM was carried out in two steps. In the first step, the PRM for medical device software life cycle processes described in IEC 62304 was developed, entitled IEC 80002-3 which is currently under national ballot at IEC. This PRM was a result of an integration of requirements from ISO/IEC 12207 and IEC 62304 following the guidelines for process descriptions set forth in ISO/IEC 24774.

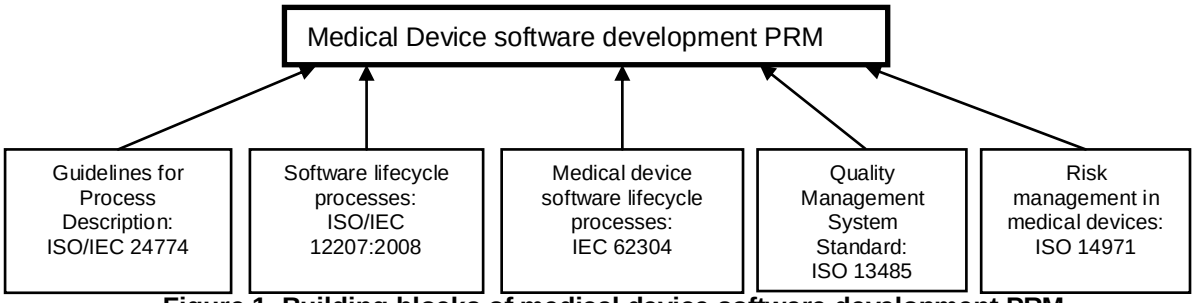

Figure 1. Building blocks of medical device software development PRM

The mapping of the requirements from the two different international standards aims at integrating the varying underlying requirements into a more abstract set of PRM-based requirements which can be applied in the development of a medical device software development PAM. In this section, we outline the approach to mapping those processes that have distinguishable one-to-one mappings from IEC 62304 to ISO/IEC 12207. With the exception of the IEC 62304 risk management process, the majority of the IEC 62304 processes are well mapped to the ISO/IEC 12207 software life cycle processes. The essential difference for many of these processes is that the safety critical activities are embedded throughout the IEC 62304 software life cycle processes.

In conducting process mappings for the directly corresponding processes, we applied the systematic approach of constant comparison and memoing as described by Grounded Theory. Constant comparison is an iterative process of data integration where the dimensions and the properties specific to data are specified [19]. This iterative process is supported by keeping memos that are a written record of analyses. Memoing "forces the analyst to think about the data and it is in thinking that analysis actually occurs" [20].

In this requirements integration activity, we conducted several iterations of constant comparison and memoing. In order to increase the validity of the mapping, a formal independent mapping of the individual processes was performed by two experienced software process researchers.

The first iteration took place when one of the researchers conducted an initial comparison of the two processes (ISO/IEC 12207 and IEC 62304). The result of this first iteration was a proposed mapping along with detailed memos that capture the reasoning behind the proposed process mappings. The second researcher then conducted a review of the first comparison and took notes for the underlying reasons for agreeing or disagreeing with the first researcher. The researchers then reviewed all the data together in the third iteration of constant comparison and they memoed the review results while finalising the requirements integration. Additional ideas and propositions of data integration that occured while the researchers finalised the comparison were again memoed and reviewed by both researchers. Iterative cycles of constant comparison were undertaken until the researchers had no conflict (or no new suggestions to the agreed requirements integration). Constant comparison is a systematic approach to analysing and integrating the process requirements that are written using different terminology and concepts [19]. Memos permit the tracking of justifications for process integration. The memos are also crucial in the validation of the PRM as the reviewers will be able to follow the data analysis and integration process in great detail.

At the time of submitting this paper, the PRM developed in the first step has been approved for publication by the IEC national bodies as IEC DTR 80002-3: Process Reference Model for Medical Device Software Life Cycle Processes (IEC 62304). IEC TR 80002-3 will be published in the middle of this year.

In the second step of the PRM development, IEC 80002-3 was extended with medical device regulatory standards that medical device software development organisations have to adhere to. The requirements from the international standards ISO 14971 and ISO 13485 where then analysed and the requirements that were not yet in the PRM were then integrated into it. ISO 13485 is a Quality Management System standard setting the requirements for regulatory purposes. ISO 14971 describes the application of Risk Management to medical devices. Both of these standards are mandatory for medical device software organisations. To have a comprehensive medical device software PRM, the relevant requirements from these two mandatory standards were included. The approach of integrating requirements from different standards was carried out similarly to the one described above with iterations of reviews by experts until there were no more contradictions in the experts' proposals for integration.

The resulting medical device software development PRM describes processes that could be grouped into three – the system life cycle processes, the software life cycle processes and the supporting processes. ISO 13485 requirements are primarily related to the system level processes which were derived from ISO/IEC 12207 in the first PRM development step. ISO 14971 maps mostly to the Software Risk Management process described also in IEC 62304.
ISO 13485 sets the requirements for Quality Management System (QMS) for Medical Devices. These requirements were integrated into the Medical Device Software Development PAM through relevant new Process Outcomes where no corresponding ones already exist, or as additional details in Base Practices where corresponding Process Outcomes already existed. Some of the QMS requirements target higher Capability Levels, in which case the requirements were related to Generic Practices PA 2.1 (e.g. on allocating resources) or to PA 2.2 (e.g. on documentation) on Capability Level 2. The outcomes or base practices derived from ISO 13485 were highlighted to visualize the source standard. This would then allow detailed feedback to companies in their compliance to the specific standards as a result of process assessment.
ISO 14971 distributes the risk management related requirements across all software life cycle processes. To avoid major duplication, the risk management requirements were kept only in the Software Risk Management process of the main body of the PAM. Instead of distributing these requirements across life cycle processes, a table was added in the Annex of the PAM that lists the specific risk management requirements for each software life cycle process. These requirements need to be added to the selected software life cycle processes for process assessment in the case where the process assessment will not include the Software Risk Management process. The table in the Annex provides the Outcomes from ISO 14971 and their corresponding Base Practice texts from IEC 80002-1 as well as the suggested location in the list of already existing Outcomes in each of the software life cycle processes.
As a result of the integration activities described above, the medical device software PRM consists of 25 processes in the three sets of software life cycle processes, software support processes, and system life cycle processes. In the following section we describe how this PRM was extended with additional elements for medical device software development process assessment model (PAM). This PAM will allow the evaluation of software and systems development processes against all the medical device standards mentioned above.

3 Development of the PAM

The aim of the Medical Device Software Development PAM is to provide a comprehensive model for assessing the software and systems development processes against the widely required medical device regulations, standards and guidelines that a software development organisation in the medical device section has to adhere to. Medical Device Software Development PAM has two dimensions - process dimension and capability dimension. Process dimension lists three groups of processes from various models and standards specified below, i.e. systems life cycle processes, software life cycle processes, and support processes. Each process is described in terms of a Process Name, Process Purpose, Process Outcomes, Base Practices, Work Products and Work Product Characteristics. Process Outcomes are the normative requirements within a process, the achievement of which will allow satisfying the Process Purpose statement. Base Practices are informative activities that illustrate one possible way (workflow) to achieve the corresponding Process Outcomes. Work Products are artefacts that are either produced or used by the Base Practices, both support the achievement of the Process Outcome. Each Work Product is further described in terms of its content called Work Product Characteristics. In the case of the medical device software development PAM, some of these Work Products are normative as they are based on the requirements derived from IEC 62304, ISO 14971 or ISO 13485. Similarly, their content may have been specified in these standards, and if that is the case, this information has been carried forward to Work Products Characteristics of the PAM.
Medical Device Software Development PAM includes information from the following standards and models. First, the baseline PAM is built upon the integrated model of IEC 62304 (Medical device software life cycle processes) and ISO/IEC 12207 (Software and Systems life cycle processes) – the PRM for IEC 62304, i.e. IEC 80002-3 and the QMS and Risk Management requirements added in the second step of the PRM development from ISO 13485 and ISO 14971, respectively. The Process Outcomes were derived from these four standards resulting in the PRM for Medical Device Software Development as described in the previous section. This PRM was extended with corresponding Base

Practices, the process implementation steps that result in the achievement of the outcomes, from IEC 62304, ISO/IEC 15504-5 and IEC 80002-1 (for ISO 14971).
As mentioned previously, IEC 80002-1 provides guidance on the application of ISO 14971 (Risk Management) to Medical Device Software. This guidance information was added to the Base practices to correspond to the requirements from ISO 14971 described in the normative part of the model illustrating a way for achieving these requirements. Most of the information from IEC 80002-1 was integrated into the PAM through Software Risk Management process Base Practices.
Additional base practice information was then derived from the FDA guidance documents on Premarket Submission [21], Software Validation [22], and Off-The-Shelf (OTS) [23] software. In most cases, this guidance was integrated into the existing processes through adding onto the existing Base Practices and Work Products or adding new Base Practices. Software Validation is not described in a separate process in IEC 62304 but as the FDA guidance provides the current best practices for software development in medical device domain, this area should also be considered. Additional processes of Software Validation, Software Installation and Software Acceptance Support were therefore added from ISO/IEC 12207 to satisfy the requirements of the FDA Guidance Document on Software Validation. Best practices from the FDA guidance documents were then analysed and iteratively integrated into these three processes.
Figure 2 below describes the different sources for the medical device software development PRM and PAM. The medical device software development PRM is based on IEC 62304, ISO/IEC 12207, ISO 14971 and ISO 13485. The PAM then extends this PRM with base practices and work products, some of the latter also being normative as they are described in IEC 62304, ISO 14971 or ISO 13485 as requirements. Where process outcomes are derived from ISO/IEC 12207, their corresponding base practices and work products are derived from ISO/IEC 15504-5. Where process outcomes are derived from ISO 14971, their corresponding base practices are derived from IEC 80002-1. In addition to these sources, FDA guidance on premarket submissions, software validation and off-the-shelf software have been added to the informative base practices where appropriate.

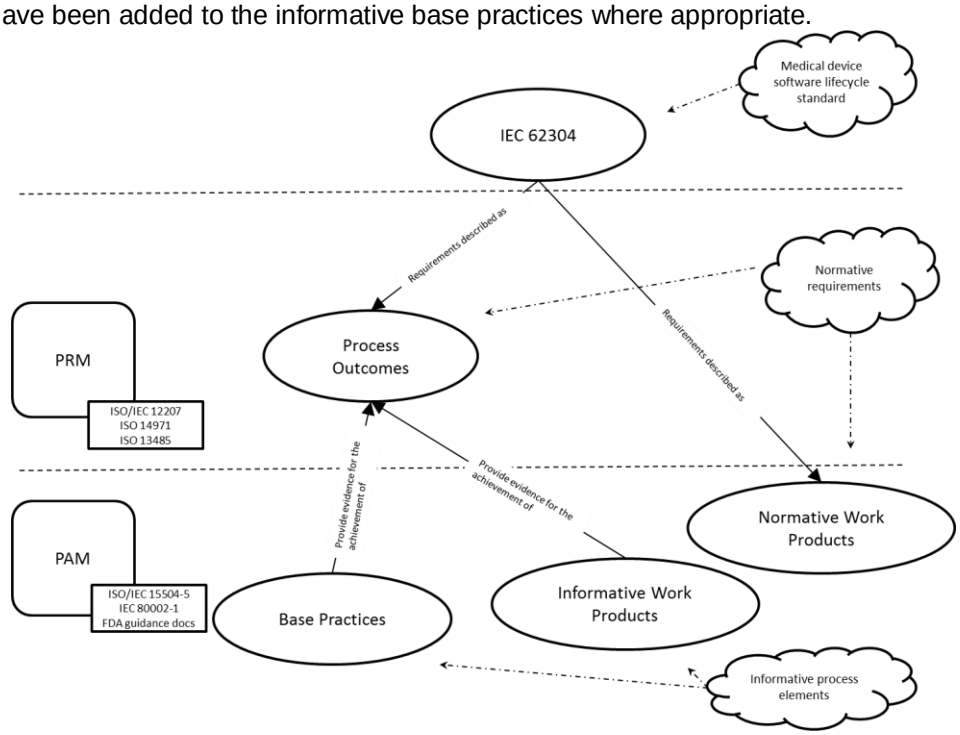

Figure 2. Normative and informative elements of the medical device software development PRM and PAM

Capability dimension of the Medical Device Software Development PAM is derived directly from ISO/IEC 15504 together with the Capability Levels, Process Attributes, Generic Practices, Generic Resources and Generic Work Products.

4 Discussion

One of the biggest challenges in integrating requirements and informative information from various sources is to do with the structure and terminology used in standards. Although ISO/IEC 24774 describes a uniform structure for process models, this has not been widely adopted quite yet. Some of the standards list their requirements in activities distinguishable as requirements only by the verb used in that sentence, e.g. “shall”, “should” or “are”.
Natural language offered another challenge in the integration of requirements from various sources. Terms like “define”, “identify”, “document”, “record”, and “establish” have each got a different meaning in some standards but in other standards they can be all be summed up under the term “establish”. It took many iterations of constant comparison before reaching a result that both reviewers were satisfied with.
The terminology of risk management in medical device standards has a different meaning from risk management in the generic software development standards where the risks are mostly related to project budget and schedule. In generic software development standards, safety engineering and safety management correspond to the product risk management central to any safety critical domain software development. Deciding on the terminology to adopt in the single best practice framework has been a great challenge. On one hand, the terminology should be the one that the safety critical domain experts use every day in their work as they will be applying the framework. At the same time, the terminology should be comprehensible for software developers across different domains allowing the latter to move into a safety critical software development.
Another significant challenge in integrating various requirements from different sources is the interface between systems and software levels. This is a challenge often faced in embedded software development. In our PAM, there are both system and software life cycle processes and for embedded software development, the interface between these levels should be very strong. One way to strengthen the interface is to trace requirements from users throughout the development to validation ensuring that the user gets exactly what he or she needed. In order to further strengthen the interface between the software and system level development processes, this traceability should be bilateral, meaning that not only should you be able to trace the user requirements through system and software requirements specification to the final product, but you should also be able to trace every feature of the product and software item back to the user needs.

5 Conclusions and Future Works

The medical device software development domain is filled with regulations that software development organisations need to comply with in order to market their products. In this paper we have described these regulatory standards together with the generic software development standards, further detailing how we have integrated requirements from both into a best practice framework.
This framework can be used for medical device software development process assessments on both system and software level. The framework provides visibility of compliance of the organisation’s processes with the requirements from all the source standards of the framework. This will allow the medical device manufacturer to select the supplier that focuses on the improvement of their medical device software development and adopts the best practices derived from the set of required standards.
At the time of submitting this paper, the framework is being validated in industry. The framework has been piloted in the first medical device companies with the aim to gather feedback from medical device software developers and to improve the framework based on this feedback before its official launch later this year.

Acknowledgment. This research is supported by Enterprise Ireland and the European Regional Development Fund (ERDF) under the National Strategic Reference Framework (NSRF) 2007-2013, grant number CF/2012/2631, and in part by the Science Foundation Ireland (SFI) Stokes Lectureship Programme, grant number 07/SK/I1299, and the SFI Principal Investigator Programme, grant number 08/IN.1/I2030 (the funding of this project was awarded by Science Foundation Ireland under a co-funding initiative by the Irish Government and European Regional Development Fund).

6 Literature

1. Feiler, P. and W. Humphrey *Software Process Development and Enactment: Concepts and Definitions*. 1992.
2. DeMarco, T. and B. Boehm, *The agile methods fray.* IEEE Computer, 2002. **35**(6): p. 90-92.
3. FDA. *Chapter I - Food and drug administration, department of health and human services subchapter H - Medical devices, Part 820 - Quality system regulation*. [cited 2013 15.05]; Available from: http://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfcfr/CFRSearch.cfm?CFRPart=820.
4. European Commission, *Directive 93/42/EEC of the European Parliament and of the Council concerning medical devices*, in *OJ o L 247 of 2007-09-21*. 1993: European Commission, Brussels, Belgium.
5. European Commission, *Council directive 90/385/EEC on active implantable medical devices (AIMDD)*. 1990: Brussels, Belgium.
6. European Commission, *Directive 98/79/EC of the european parliament and of the council of 27 october 1998 on in vitro diagnostic medical devices*. 1998: Brussels, Belgium.
7. European Commission, *Directive 2007/47/EC of the European Parliament and of the Council concerning medical devices*, in *OJ no L 247 of 2007-09-21*. 2007, EC: Brussels, Belgium.
8. ISO, *ISO 13485: Medical Devices - Quality Management Systems - Requirements for Regulatory Purposes*. 2003, ISO: Geneva, Switzerland.
9. ISO, *ISO 9001:2000 - Quality Management Systems - Requirements*. 2000: Geneva, Switzerland.
10. IEC, *IEC 62304: Medical Device Software - Software Life-Cycle Processes*. 2006, IEC: Geneva, Switzerland.
11. ISO/IEC, *ISO/IEC 12207:1995 - Information Technology - Software Life-Cycle Processes*. 1995, ISO/IEC: Geneva, Switzerland.
12. ISO/IEC, *ISO/IEC 12207:2008 - Systems and Software Engineering - Software life cycle processes*. 2008, ISO/IEC: Geneva, Switzerland.
13. ISO/IEC 15504-2, *Information technology - Software process assessment - A reference model for processes and process capability*, in *15504*. 2004.
14. ISO, *ISO 14971 - Medical Devices - Application of Risk Management to Medical Devices* 2009, ISO: Geneva, Switzerland.
15. IEC, *IEC TR 80002-1 - Medical Device Software - Part 1: Guidance on the Application of ISO 14971 to Medical Device Software*. 2009, IEC: Geneva, Switzerland.
16. IEC, *IEC 60601-1 - Medical electrical equipment – Part 1: General requirements for basic safety and essential performance* 2005, IEC: Geneva, Switzerland.
17. IEC, *IEC 62366 - Medical devices - Application of usability engineering to medical devices*. 2007, IEC: Geneva, Switzerland.
18. ISO/IEC, *ISO/IEC 24774 - Systems and Software Engineering - Life Cycle Management - Guidelines for Process Description*. 2010: Geneva, Switzerland.
19. Clarke, P. and R.V. O'Connor, *The situational factors that affect the software development process: Towards a comprehensive reference framework.* Information and Software Technology, 2012. **54**(5): p. 433-447.
20. Corbin, J. and A. Strauss, *Basics of Qualitative Research*. 2008, Thousand Oaks, CA, USA: Sage Publications.
21. FDA, *Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices*. 2005. p. 20.
22. FDA, *General Principles of Software Validation; Final Guidance for Industry and FDA Staff*. 2002. p. 43.
23. FDA, *Guidance for industry, FDA reviewers and compliance on - Off-The-Shelf Software Use in Medical Devices*. 1999. p. 26.

Author CVs

Dr. Marion Lepmets

Dr. Marion Lepmets is a Senior Research Fellow at the Regulated Software Research Centre in Dundalk Institute of Technology and a member of the Irish Software Engineering Research Centre (Lero). Her PhD and post-doc research focused on the impact of process model implementation and of process assessment on process improvement success, respectively. Her current research is about the development of the process reference and process assessment models for medical device software development. She is also involved in the development of software engineering standards in the International Standardization Organization (ISO/IEC JTC1 SC7).

Dr. Paul Clarke

Dr. Paul Clarke is a Research Manager in the Regulated Software Research Centre based at DkIT and a Research Fellow with Lero – the Irish Software Engineering Research Centre. Primary among Paul's present research interests is the establishment of a best practice framework for medical device software development.

Dr. Fergal McCaffery

Dr Fergal Mc Caffery is a Science Foundation Ireland (SFI) Principal Investigator. He is a Senior Lecturer in the Department of Computing and Mathematics, Dundalk Institute of Technology (DkIT). He is Director of the Regulated Software Research Centre in DkIT and the Medical Device Software Engineering competency area in Lero. He has been awarded SFI funding through the Stokes Lectureship, Principal Investigator and CSET Programmes to research the area of medical device software. He has published over 150 peer-reviewed conference and journal papers and is on the editorial board/programme committee for a number of leading software engineering conferences and journals.

Anita Finnegan

Anita Finnegan is a Senior Software QMS Researcher in the Regulated Software Research Centre based at DkIT and a member of Lero – the Irish Software Engineering Research Centre. Anita's primary focus is establishing the relationship between QMS requirements and software development requirements. Her PhD research addresses the development of a framework for the security assurance of medical devices using cybersecurity assurance cases.

Alec Dorling

Alec Dorling is a member of the research team at the Regulated Software Research Centre in Dundalk Institute of Technology and a member of the Irish Software Engineering Research Centre (Lero). He is the ISO convener of the ISO/IEC 15504 series and ISO/IEC 330xx family of standards on Process Assessment, and the international project leader of SPICE (Software Process Improvement and Capability dEtermination). He has held key positions in software engineering at both national and European levels at IVF's Centre for Software Engineering in Sweden, IT Research and Technology Transfer Centre at the University of Boras in Sweden, European Software Institute in Spain and the National Computing Centre in the UK. He is on the editorial board/programme committee for a number of leading software engineering conferences and journals.